Mask-Edge Connectivity

Wilkinson, Michael H.F.; Oosterbroek, Jaap

Published in:
Proceedings of the 21st International Conference on Pattern Recognition, ICPR 2012

Publication date:
2012

21st International Conference on Pattern Recognition (ICPR 2012)
November 11-15, 2012. Tsukuba, Japan

Mask-Edge Connectivity: Theory, Computation, and Application to Historical Document Analysis

Michael H.F. Wilkinson and Jaap Oosterbroek
Johann Bernoulli Institute, University of Groningen,
P.O. Box 407, 9700 AK Groningen, The Netherlands
m.h.f.wilkinson@rug.nl; *jaap.oosterbroek@gmail.com*

Abstract

In this paper a new connectivity model is introduced which allows combined clustering and partitioning of structures without distortion, in contrast to mask connectivity. An algorithm to compute morphological attribute filters based on Max-Trees for this new form of connectivity is presented. It is shown that the new form of connectivity is effective in clustering diacritics together with the appropriate letters in historical documents, whilst separating letters from different lines in a single Max-Tree algorithm.

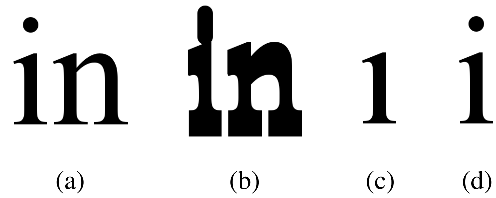

Figure 1. Mask connections: (a) text; (b) connectivity mask; (c) "i" detected by 4-connectivity, (d) "i" detected by mask connectivity using (b)

1. Introduction

Connected attribute filters [1, 6] are unique in that they filter images based on the properties or *attributes* of connected components rather than predefined neighbourhoods of pixels. As such, they rely on a notion of connectivity, which in mathematical morphology is defined using connectivity classes [7]. A connectivity class is the family of all subsets of the image domain which are considered connected. Usually, this consists of the collection of all path-connected sets deriving from the 4 or 8 neighbour relation in 2D images. By manipulating the notion of connectivity, we can obtain task-specific perceptual grouping schemes. We might, e.g., want to consider the dot over an "i" as part of the same connected set as the remainder of the letter, as is shown in Fig. 1. Likewise, accents over characters can be considered connected to the letter below it.

A very general ways of doing this is through mask connectivity [3]. This uses a *mask* image to determine what is and is not connected. Though this works very well for clustering, there are problems when we want to separate touching objects. In this case, we will have to distort at least one of the objects, by setting foreground pixels in the original image to background in the mask [9]. Inspired by [2], where watersheds are computed on weights assigned to edges, we propose to add edge weight information to the mask connectivity scheme. In this hybrid scheme, clustering is performed by the mask image, whereas manipulating the edge weights allows splitting regions without deformation.

The paper is organised as follows. First a theoretical background is given, and the theory is extended to mask-edge connectivity. After this, we discuss how the algorithm for the Max-Tree [5] given in [10] can be adapted to this connectivity. Finally, an experiment is performed on the problem of character segmentation and recognition in historical documents.

2. Connectivity and Connected Filters

We will first consider the binary case for simplicity, and move to the grey-scale case afterwards. In the binary case images X are considered subsets of some universal set E (the image domain) which is generally a subset of $\mathbb{Z}^2$. The power set of E is denoted $\mathcal{P}(E)$. Foreground pixels are members of X background pixels members of $X \setminus E$. A *connectivity class* or *connection*

$\mathcal{C} \subseteq \mathcal{P}(E)$ [7] is any family of subsets of E such that

1. $\emptyset \in \mathcal{C}$, and for all $x \in E$, $\{x\} \in \mathcal{C}$,
2. for any $\{C_i\} \subseteq \mathcal{C}, \bigcap_i C_i \neq \emptyset \Rightarrow \bigcup_i C_i \in \mathcal{C}$

Thus, the empty set and all singletons (pixels in the discrete case) are considered connected. Furthermore, if the intersection of any collection of connected sets $\{C_i\}$, is not empty (i.e. they overlap), their union is also connected. Any image can be decomposed into *connected components*, which are connected sets of maximal extent. These can be extracted by means of *connectivity openings* Γ_x. For a given location $x \in E$, $\Gamma_x(X)$ returns the connected component of X to which x belongs, if $x \in X$ and $\emptyset$ otherwise.

Attribute filters [1] can be derived from these connectivity openings by means of trivial filters Ψ_Λ, given by

$$\Psi_\Lambda(C) = \begin{cases} C & \text{if } \Lambda(C) \\ \emptyset & \text{otherwise} \end{cases} \tag{1}$$

in which Λ is a selection criterion, and C some connected set. Criterion Λ usually has the form

$$\Lambda(C) = \big(\mu(C) \geq \lambda\big) \tag{2}$$

with $\mu(C)$ some measure or *attribute* of C, such as area, and λ the *attribute threshold*. Many examples of attributes are given in [1, 5]. The attribute filter Ψ^Λ is then

$$\Psi^\Lambda(X) = \bigcup_{x \in X} \Psi_\Lambda(\Gamma_x(X)) \tag{3}$$

which returns the union of all connected components for which Λ holds. In the grey scale case, we can in principle compute attribute filters by thresholding the image at all levels, performing the binary filters at each level, and superimposing the results. In practice, more efficient methods are used [1, 5, 10].

Connectivity in image analysis is usually based on 4 or 8 pixel adjacency relationships. The image is implicitly seen as a graph, with the pixels as the vertices, each connected to 4 or 8 neighbours. Using the notion of path or arc wise connectivity, any set for which there exists an unbroken path from one pixel to any other pixel is a member of the base connectivity class $\mathcal{C}$. Fig. 1(a) has three foreground components using $\mathcal{C}$, one of which is shown in Fig. 1(c), by only allowing paths consisting of edges connecting foreground pixels.

In mask connectivity, we only allow paths consisting of edges that connect foreground pixels in the *mask* image. Fig. 1(b) shows a mask obtained by dilation with a vertical line of 50 pixels. This connects the "i" with the dot above, and yields two connected components, rather than three, as shown in Fig. 1(d). Formally, we obtain these connected components using a new connectivity opening Γ_x^M, which is defined as

$$\Gamma_x^M(X) = \begin{cases} \Gamma_x(M) \cap X & \text{if } x \in X \cap M, \\ \{x\} & \text{if } x \in X \setminus M, \\ \emptyset & \text{otherwise.} \end{cases} \tag{4}$$

with M the mask image. This means that if $x \in M \cap X$, connected components are determined by performing a regular connectivity opening on M, and intersecting it with the original image. Foreground pixels in X which are background in M are treated as singletons. Attribute filters using this connectivity are defined by replacing Γ_x by Γ_x^M in (3).

Clustering disconnected entities can be carried out effectively using this scheme, but if, e.g., two letters touch, the only way to separate them is by setting foreground pixels in the original image to background in the mask. We always end up with one or more singleton pixels in this case, and distort one, or both of the remaining objects. This is because only by setting a mask *pixel* to background we can tell the algorithm that a particular *edge* should be removed. What we really want to do is to remove edges without affecting the status of pixels. We can achieve this by modelling the edges between pixels explicitly, rather than implicitly. In the binary case, we no longer model images as sets of the vertices of the graph, but as sets of vertices and edges, following, e.g., [2]. In the next section we move to the grey scale case, and show how mask-edge connectivity can be implemented in that case.

2.1. Grey-Scale Connected Filters

Grey scale filters can be computed using threshold superposition, in which the image is thresholded at each grey level, a binary filter applied to all binary images, and the results stack up into a grey-scale result. This is done efficiently through Max-trees or Min-trees [5]. Each node in the tree represents a *peak component* or connected component of a threshold set of the image, see Figure 2. In 4-connectivity, pixels at level h are connected at that level if there is a 4-connected path through pixels at level $\geq h$ in the original image. In mask connectivity, the same holds, but for mask pixels. In mask-edge connectivity, the path must be through mask pixel *and edges* of level $\geq h$. In a Max-Tree regional maxima form the leaves, whereas in a Min-Tree the regional minima are the leaves.

In [3], the algorithm from [5] was adapted to compute Max-Trees for mask connectivity. This is know as the dual-input Max-Tree algorithm. A new algorithm which can readily compute both the regular and

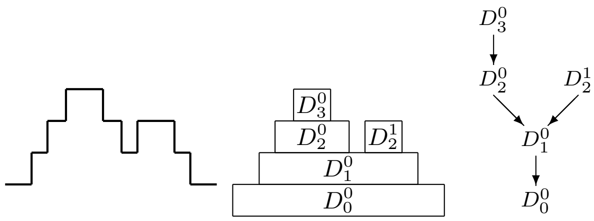

Figure 2. A 1-D signal f (left), the corresponding peak components (middle) and the Max-Tree (right).

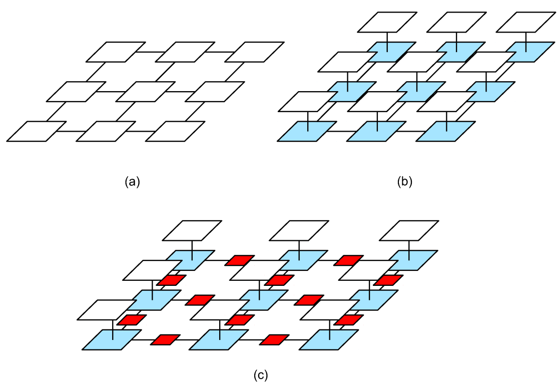

Figure 3. Three forms of connectivity: (a) regular; (b) mask; (c) mask-edge. White nodes are original pixels, blue nodes mask pixels, red nodes are edges.

dual-input Max-Tree was presented in [10]. Instead of adapting the algorithm for different connectivities, this approach adapts the image representation.

During the flood-fill stage of algorithms from [5,10], a function `getNeighbours` returns the neighbours of the current pixel. In normal 4-connectivity, this returns the 4 neighbours as shown in Figure 3(a), except at the image borders. In mask based connectivity, the regular pixels have just one neighbour, i.e., the corresponding pixel in the mask, whereas the mask pixels have 5 neighbours: four mask pixels and the corresponding image pixel, see Figure 3(b). Furthermore, when computing properties such as area, only the original image pixels may contribute. This is done by initializing the area of the original pixels to one, and that of the mask pixels to zero. A similar approach can be made for other moment-based attributes. Just these simple changes to the neighbourhood relationship and initialization allow the algorithm from [10] to compute both types of connectivity. For details see [10].

Table 1. Recall rates

connectivity	size	weighted recall	COI recall
4	-	0.81 (0.78-0.85)	0.25 (0.10-0.35)
mask	4	0.80 (0.78-0.84)	0.31 (0.27-0.41)
mask	8	0.81 (0.78-0.86)	0.63 (0.49-0.73)
mask	15	0.73 (0.67-0.80)	0.74 (0.67-0.88)
mask-edge	4	0.80 (0.73-0.86)	0.30 (0.24-0.41)
mask-edge	8	**0.82 (0.77-0.88)**	0.61 (0.50-0.73)
mask-edge	15	0.75 (0.65-0.85)	**0.74 (0.53-0.91)**

To include mask-edge connectivity, we extend the above strategy, by inserting a third class of "pixel" in the data structure, namely those representing the edges, which now have a grey level, as shown in Figure 3(c). Original image pixels are connected as before, but mask pixels are now connected to four edges, and these edges are in turn connected to two mask pixels. Obviously, edges have zero area. The algorithm of [10] can compute mask-edge connected Max-Trees by these changes to `getNeighbours`, and the initialization.

3. Experiments

As data set we used scanned pages from the *Rerum Frisicarum Historia* of Ubbo Emmius (1596). Images were inverted so letters have high grey levels. A ground truth letter segmentation and classification was obtained by manual annotation. We implemented the algorithm in Java, using normalised central moments as vector attribute [8] for each Max-Tree node. Nodes were then classified using a nearest-neighbour classifier with respect to a ground-truth training set. Background details were removed by the k-flat method from [4].

The mask was computed by dilation with a vertical line structuring element (SE) of varying lengths. Edge weights were generally set at 255 (always connect), though they could equally be set to the minimum grey value of the neighbouring mask pixels. The exceptions are the edges between two consecutive lines of text. We first summed the pixel grey levels in the x direction obtaining a profile. Minima in this profile should correspond to line separations. Edges between pixels on either side of such a separation were set to 0.

Table 1 shows the results. The first column lists the connectivity used. The second indicates the size of the structuring element used. The third column shows the recall weighted by class frequency. The final column shows the recall score for all *characters of interest* (COI), i.e. those with accents or dots. Recalls are shown as the mean value with the 1st and 3th quartile

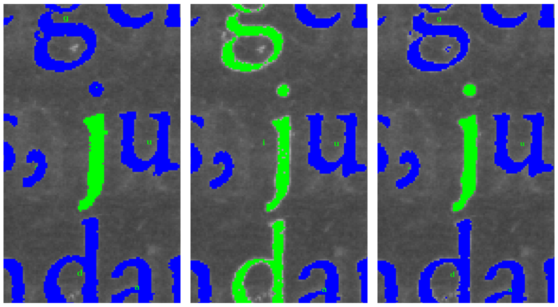

Figure 4. Segmentation results: Left to right, segmentation using standard connection, mask connection, and mask-edge-based connection.

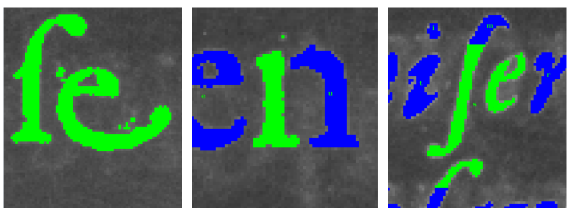

Figure 5. Segmentation problems: Left to right: inappropriate clustering; horizontal splitting; bad line boundary.

in parentheses. Bold scores are the highest in their column. Given the rarity of the COI, precision as a statistic was considered less meaningful.

The results show that improvements are obtained in terms of recall by using dilation. This is mainly due to improvements in detecting the COI. For larger SE sizes we have some degradation of overall performance, due to incorrect linking of characters. This effect is smaller in mask-edge connectivity, which has a slight edge over mask connectivity.

These effects can be seen in detail in Figure 4, which shows the same page section containing the letter 'j', as it is segmented by the three different Max-Trees. Regular connectivity does not link dot to the letter 'j'. Figure 4(b) shows incorrect linkage of letters to the 'j' for large SE in mask connectivity. Figure 4(c) finally shows the correct segmentation using mask-edge connectivity.

The chosen mask and mask-edge method also introduces a few problems. Figure 5(a) shows unintentional clustering of two characters, which happens most with the 'f' and 's' found in old print. Figure 5(b) shows a split in the letter 'n'. This is in part due to the preprocessing method used [4]. Here horizontal clustering would be needed. Finally, figure 5(c) shows the main problem of mask-edge segmentation in this context. Our line splitting algorithm has problems finding the separations between lines, because it assumes a straight, horizontal split. These splits are probably the main reason mask-edge connectivity does not outperform mask connectivity by a larger margin.

4. Conclusions

Mask-edge connectivity can be implemented efficiently using the algorithm in [10]. The computational cost is about twice that of mask connectivity, due to the increased number of nodes in the graph. Mask-edge connectivity also shows promise in performing perceptual grouping more effectively than mask connectivity. Having said that, we clearly need a better line separation method, using e.g. a minimum-cost path algorithm. Finally, a better, more adaptive clustering approach is needed to repair erroneously split letters.

References

[1] E. J. Breen and R. Jones. Attribute openings, thinnings and granulometries. *Comp. Vis. Image Understand.*, 64(3):377–389, 1996.

[2] J. Cousty, G. Bertrand, L. Najman, and M. Couprie. Watershed cuts: Minimum spanning forests and the drop of water principle. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(8):1362–1374, Aug. 2009.

[3] G. K. Ouzounis and M. H. F. Wilkinson. Mask-based second generation connectivity and attribute filters. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29:990–1004, 2007.

[4] G. K. Ouzounis and M. H. F. Wilkinson. Hyperconnected attribute filters based on k-flat zones. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(2):224–239, 2011.

[5] P. Salembier, A. Oliveras, and L. Garrido. Anti-extensive connected operators for image and sequence processing. *IEEE Trans. Image Proc.*, 7:555–570, 1998.

[6] P. Salembier and M. H. F. Wilkinson. Connected operators: A review of region-based morphological image processing techniques. *IEEE Signal Processing Magazine*, 26(6):136–157, 2009.

[7] J. Serra. Connectivity on complete lattices. *J. Math. Imag. Vis.*, 9(3):231–251, 1998.

[8] E. R. Urbach, N. J. Boersma, and M. H. F. Wilkinson. Vector-attribute filters. In *Mathematical Morphology: 40 Years On, Proc. Int. Symp. Math. Morphology (ISMM) 2005*, pages 95–104, Paris, 18-20 April 2005.

[9] M. H. F. Wilkinson. Attribute-space connectivity and connected filters. *Image Vis. Comput.*, 25:426–435, 2007.

[10] M. H. F. Wilkinson. A fast component-tree algorithm for high dynamic-range images and second generation connectivity. In *Proc. Int. Conf. Image Proc. 2011*, pages 1041–1044, 2011.